JPRS-UCC-89-002

29 MARCH 1989

FOREIGN BROADCAST INFORMATION SERVICE

# JPRS Report

# Science & Technology

## USSR: Computers

JPRS-UCC-89-002

29 MARCH 1989

# SCIENCE & TECHNOLOGY

# USSR: COMPUTERS

## CONTENTS

APPLICATIONS



NETWORKS

UDC 62-192:681.3.06:652.011.012.56

# Reliability of the Software Inventory for Automated Process Control Systems

18630234a Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan-Feb 88 (manuscript received 22 Apr 86, after correction 6 Feb 87) pp 27-32

[Article by V. P. Zhabeyev]

[Text] Formulation of Problem

Software has been reclassified as a product of industrial significance. Thus, the classes and groups of the industry-wide system of standards that previously dealt with the development, production and use of hardware has now been extended to include inventoried software, i.e., programs on data medium that are the product of commercial production (GOST [All-Union State Standard] 19.004-80). Of most relevance are the 4th class of standards--product quality indices, the 15th--development and organization of production, and the 27th--the "Reliability in Engineering" standards.

While acknowledging the fund mental differences that exist bewteen methods of producing, controlling, testing and using hardware versus software, we will adopt the following thesis: The quality must be based on the indices for both hardware and software system existing, thoroughly-tested system for controlling the quality and testing of products, with its unififed methodological and methodology and standards [1].

A representative (italicized by the author here and below) structure of measurable quality indices for the reliability of inventoried software is laid out in this article. This structure provides a single value for the reliability of inventoried software, while taking maximum advantage of the existing system of technical documentation standards. The principal requirements of facilities for the automated measuring of quality indices for inventoried software are also specified in this article.

## Basis for Set of Axioms Used

A general definition of the concept "reliability" is given in GOST 27.002-83, "Reliability in Engineering. Terms and Definitions," and as it relates to automated process control systems, in GOST 24.701-83, "ASUTPs. Reliability. Basic Assumptions."

The following axioms are fundamental to these standards:

A1--a product can be in either serviceable or non-serviceable condition. Although the term "serviceability" is introduced in GOST 27.002-83, in this paper it is regarded in accordance with [2] as a broader property than reliability.

A2--a product maintains the values of all its parameters over time and within set limits.

The following components of reliability are a logical extension of axioms A1 and A2: failure-free performance (B), longevity (D), maintainability (R) and consistency (S). In turn, the nature of each component is based on one or more components of the next level of the hierarchy.

Failure-free performance:

B1--serviceable condition is maintained for a certain period of time.

B2--serviceable condition is maintained for a certain accrued operating time.

Longevity:

D1--the product is considered serviceable until the onset of critical condition, i.e., the condition beyond which further use is impermissible or inadvisable (GOST 27.002-83). For example, technical and economic factors, the consequences of failures and critical condition, etc., can be used as indications of failures and critical condition in inventoried software (GOST 27.103-83, "Reliability in Engineering. Criteria for Failures and Critical Condition. Basic Assumptions").

Maintainability:

R1--the product must be adapted for the prevention and detection of the causes of failure and defects and for the maintenance and restoration of serviceable condition.

Consistency

S1--the product maintains the values of its indices for failure-free performance, longevity and maintainability both during and after storage.

S2--the product maintains the values of its indices for failure-free performance, longevity and maintainability both during and after transport.

The validity of these axioms for defining the software quality indices is a subject of fundamental dispute. The following specific features of software are advanced in [3-6] as the main arguments against using the existing system of technical documentation standards, or at least its principal documents.

Hardware failures result from physical defects in individual components, the influence of the environment, operating conditions, etc. For software, failures are due to algorithm and program errors, the number and cost (consquences) of which are determined to a considerable degree by the subjectively chosen development technology.

Hardware, unlike software, is subject to physical wear and aging.

It is impossible to increase the value of a software quality index by replacing one component with another copied from the same original, since one of the conditions of copying is that the copy must be identical to the original.

A specific defect can be eliminated by replacement of the defective component with a component that is based on another algorithm or development method; but the possibility of introducing new algorithm and program errors in the process cannot be discounted.

Other characteristics of software that both support and refute the arguments advanced in [3-6] can be pointed out. In the author's opinion, this situation has resulted from attempts to use a single set of quality indices for all stages of the software life cycle. The irrationality of this approach is pointed out in [3].

Having discussed the main points of view concerning the reliability of software, and synthesizing the positive results obtained in [3-6], let us proceed to the selection and substantiation of each component of software reliability, while limiting ourselves to one stage of its life cycle: "Funding, Reproduction, and Support."

Reliability Models for Inventoried Software

For hardware, the concepts "time" (A2, B1) and "accrued operating time" (B2) are used in the most obvious, concrete sense and are essentially analogous, since accrued operating time can be examined only over a certain time period. But if a program is run repeatedly with the same input data (the determinate variant), then the same errors either will or will not occur repeatedly in the process, since there are no destabilizing factors (wear and physical aging) for software. Its serviceability will be checked for one point, L, or for a region, M, (fig 1a) defined by the set of inputs

$$Z[M] = F([X], [Y]);$$
$$X_1 \leqslant [X] \leqslant X_2, \quad Y_1 \leqslant [Y] \leqslant Y_2, \tag{1}$$

where $[\cdot]$ indicates that the corresponding value of factors X and Y is a discrete quantity belonging to the set $[X_{1,2}, Y_{1,2}]$.

If the values of the factors and the limits of the constraints in (1) are selected randomly, then it becomes possible to cover region M with discrete

points whose density is determined by the characteristics of the rule for selection of input sets and algorithms for processing them. i.e., by the testing strategy used.

The serviceability of software can be tested only for discrete points of region Z by means of a sequence of procedures of this kind (determinate and/or random), since testing the paths of all logical routes is practical only for relatively simple software, such as programs for input/output, interchange, transmission, and calculation of technical and economic indices. Testing all routes for more complicated software (application packages and libraries, compilers, operating systems and versions of them, etc.) is often infeasible from a practical and/or theoretical (algorithmic) standpoint. According to Nelson's model [7], whose argument is the number of program runs, only an estimate of reliability can be obtained in these cases:

$$R_{(n)} = \exp\left\{\sum_{m=1}^{n} \ln (1 - P_m)\right\}. \qquad (2)$$

where $P_m$ is the probability of failure when executing the m-th run of the program and $R_{(n)}$ is the probability of failure-free execution of n runs of the program. In addition, this estimate is reliable only when each run is executed with a new input set, (1).

Let us assume that in the process of testing, the software's serviceability was not tested in local region $M_1$ (fig 1a), and that the input sets, $[X_{3,4}, Y_{3,4}]$, which define this region, do in fact result in failure situations and/or failures if $I_1$ (a parameter of the tested software's reaction) exceeds a specific level--the "protective failure" level [4] (fig 1b).

Any values of the most important factors (for the specific application conditions) can be chosen as the boundary distinguishing failure situations from failures: the expenditure of physical, technical and/or time resources necessary for eliminating the consequences of a failure or failure situation, the probability of restoration of serviceable condition, the efficiency preservation factor, etc. The boundary between a software error and failure can be drawn similarly.

Another (time) representation of region Z is presented in fig 1b, from which it follows that failure situations corresponding to input sets (1) can originate only in intervals $\Delta t_1$, $\Delta t_2$ and $\Delta t_3$, and failures at moments in time $A_1$, $A_2$, $B_1$-$B_3$ and $C_2$. In this interpretation of the relationships between the factor (cf. fig 1a) and time (cf. fig 1b) representations of regions of action of input sets, the characteristics of the stream of software errors (failure rate, mean time between failures, operating time to failure, etc.) take on the meaning of a function of time, i.e., one that is analogous to the characteristics of the stream of hardware errors.

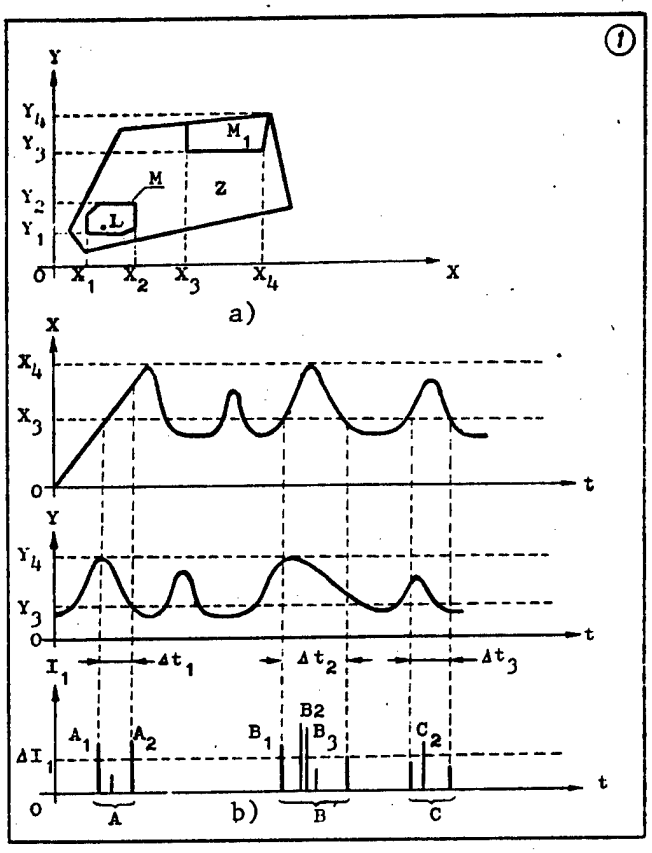


Figure 1. Forms of Representation of Signals: a--factor; b--time

As an illustration, the number of errors of a certain kind of software, $PS_i$, used by various users, $P_1$ and $P_2$, is presented as a function of time in fig 2. The streams of errors $n_i'$, $n_i''$, $n_i(\Sigma) = n_i' + n_i''$, of the software $PS_i$ that is in use have different characteristics for each case: $t_H'$ and $t_H''$ represent the time of the beginning of its use (delivery), $n_i(j)$ is the number of errors, where $j = 1, 2$, and $t_H'$, $t_1'$, ..., $t_3'$ and $t_H''$, $t_1''$, ..., $t_3''$ are the moments of appearance of errors.

Thus, for software, too, the stream of errors is determined by how it is used and the conditions under which it is applied and serviced. Under conditions of servicing we include the possibility that additional errors will be introduced in the process of eliminating old ones.

Taking into account these distinctive features of the software and the results obtained in [8], the function for the number of errors in the software $PS_i$ that is being used by a specific user has the following form:

$$E_i[\Pi_j(t)] = E_0 l^{-kt} \pm \alpha Q_0 (1 - l^{-kt}), \qquad (3)$$

where $E_0$ is the initial (for a specific $t_n$) number of errors, $Q_0$ is the rate of introduction of new errors, k is the error rate, and $\alpha$ is the degree of the influence ("weight") of the errors introduced, where $0 \leq \alpha \leq 1$.

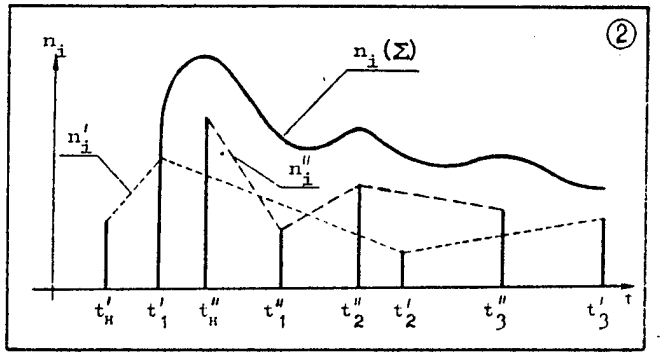


Figure 2. Number of Errors Revealed by Users, $P_1$ and $P_2$, in Process of Using $PS_i$

Relationship (3) adequately demonstrates one distinctive feature of inventoried software--the dependence of the stream of errors on conditions of operation, application and servicing. However, in the course of its application by the consumer, the use of relationship (3) encounters a number of difficulties (sometimes of a fundamental nature): It is necessary to (determine) the initial number of errors, the rate of introduction of new errors, the "weight" (consequence) of each error and the error rate. Thus, an expression that is analogous to Bernoulli's theorem and is discussed in [11] is more appropriate for solving the problems formulated in this paper:

$$R_i(\Pi_j) = 1 - \frac{n_i^{(j)}}{N}, \qquad (4)$$

where $R_i(\Pi_j)$ is the probability of failure-free operation of $PS_i$ for specific user $P_j$, N is the number of startups of $PS_i$, and $n_i^{(j)}$ is the number of successfully completed startups.

By virtue of the simplicity of gathering and obtaining this kind of data and the obviousness of relationship (4), it is easy enough for the software supplier to make use of. However, he must take into account the following features of relationship (4), which was determined by the specific software he is using, his testing strategy and the operating conditions. Failures, errors and failure situations represent a complete group of events only in the case when the input sets uniformly cover region Z, i.e., when $PS_i$ is used in all modes and under all application and servicing conditions regulated by the relevant technical documentation standards.

An analysis of relationships (2) to (4), as well as the results obtained in [1, 3 and 7-10], makes it possible to assert the following:

The reliability of inventoried software does not by its nature depend on time, but in the actual use of the software, it appears as a function of time.

The representation of the reliability of inventoried software as a function of time is used as a convenient physical interpretation based on the obvious factor that the error stream of software that is in use is a function of time, operating modes and conditions of application and servicing.

A statement concerning an improvement in the reliability of the software in use is valid only if, in the elimination of errors detected or in the improvement of individual quality indices, new errors are not introduced or theor influence is small enough to be disregarded ($\alpha \approx 0$), and if the following condition is fulfilled: The "weight" of the next error detected is smaller than that of the preceding one.

The number of errors detected is reduced, as a rule, and the reliability of the software in use is increased with an increase in the number of users, which is equivalent to an increase in the intensiveness of use of the software, and the coordinated exchange of information among the users.

Conclusion

In spite of the differences in the nature of hardware, versus software, failures (errors), the following conclusions can be drawn.

Axioms A1, A2, B1, B2 and R1 are applicable for inventoried software as a PTN [not furthered identified] product, since the reliability of inventoried software can be represented as a function of time, modes and conditions of application, servicing and repair.

The possibility of introducing significant new errors logically implies the applicability to software, too, of A1 and of the concept of critical condition (D1), since according to GOST 27.103-83 the appearance of processes that hinder the functioning of the equipment is also an indication of failure and critical condition, in addition to the indications pointed out earlier.

The idea of software as a program product, i.e., a program on data medium that is the product of commercial production (GOST 19.004-80), the expansion of the delivery area and the more extensive use by consumer of computer media all increase the probability of damage to the media during use, storage and transport. The service life of standard media is thus shortened. This situation calls for observation by the supplier of the axioms defining quality in storage (S1) and transport (S2).

The lack of a time relationship between working and conflict input sets makes it possible to organize accelerated tests of inventoried software.

Taking the above into account, as well as [1] and GOST 27.002-83, let us formulate a definition of the reliability of inventoried software: Preservation of the values of quality indices for the reliability of inventoried software can be guaranteed for an unlimited time of use and/or accrued operating time if the modes and conditions of application, servicing, repair, storage and transport meet fully the conditions and constraints specified by the technical documentation (service log, testing program and procedure, and specifications).

This definition is sufficiently constructive for the listed problems, in that it clarifies the relationships between the developer, supplier, and user. Furthermore, it does not remove responsibility from the supplier, as might appear at first glance, since according to the standards in force (OST [All-Union Standard] 26 1121-83, "Development and Set-up of Software for Production in the Instrument Making Industry," and OST 25 780-84, "Software. Procedure and Rules for Making Tests") the supplier is obliged to participate in all kinds of tests on the software, beginning with the conceptual and/or technical designs.

On the other hand this definition of the reliability of software imposes a number of completeness (thoroughness) requirements on the testing. According to GOST 20911-75, "Technical Diagnosis. Principal Terms and Definitions," verification tests and/or tests for finding defects, procedures for choosing tests inputs [11-13], and a procedure for supplying them to the input of the software being examined are required [14-16].

The approaches discussed in detail in [14-16] make it possible when developing an inventory of software to implement both determinate and stochastic testing. Stochastic testing deserves particular attention, for it is the most effective when supplying nontrivial software--application packages, in particular.

In conclusion it is necessary to point out the existence among users of fairly widespread, but up to now insufficiently formalized, qualitative methods of determining the values of indices for software reliability by means of fuzzy relationships [17, 18]. Under this approach isolated, comprehensive, or integrated quality indices are determined by means of one of two equivalent scales: the linguistic (<excellent>, <very good>, <good>, <satisfactory>, <poor>, <very poor>), or numerical (point)--from 1 to 7 (10) points. In principle other quality scales could also be used.

In spite of the subjectivity factor and the "roughness" of approximation (the use of a scale not higher than an interval scale is permitted), the method of fuzzy relationships makes it possible to obtain sufficiently representative estimates [18, 19], particularly at the initial stages in the development of software and with limited data, e.g., if the software is used for an insignificant calendar period or the number of users is limited. It should also be mentioned that the method of fuzzy relationships in no way contradicts, and is even based on, expert rating methods regulated by a group of GOSTs (23554.0-79, 23554.1-79, 23554.2-81). These state standards are the basic ones for the organization and performance of the expert rating of software before it is added to the available inventory.

BIBLIOGRAPHY


1. Kulakov, A. F. "Otsenka kachestva programm EVM" [Evaluation of Quality of Computer Programs], Kiev, Tekhnika, 1984, 167 pages.

2. Kudryashov, Yu. P. "'Serviceability' as a Defining Property of the Quality of Complex Systems," PRIBORY I SISTEMY UPRAVLENIYA, No 6, 1985, pp 12-14.

3. Lipayev, V. V. "Quality of Software, Moscow, Finansy i statistika, 1983, 263 pages.

4. Zarenin, Yu. G. "Voprosy obespecheniya nadeznhosti ASUTP" [Assuring the Reliability of ASUTPs], Moscow, Znaniye, 1983, 116 pages.

5. Dobrovskiy, V. F. and Ostianu, V. M. "Correctness and Reliability of Software" in "Abstraktnaya i strukturnaya teoriya releynykh ustroystv" [Abstract and Structural Theory of Relay Devices], Moscow, Nauka, 1982, pp 91-131.

6. Boem, B., Braun, Dzh. and Kaspar, Kh. "Kharakteristiki kachestva programmnogo obespecheniya" [Quality Characteristics of Software], Moscow, Mir, 1981, 208 pages.

7. Gorskiy, L. K., Karpovskiy, Ye. Ya. and Chizhov, S. A. "Evaluation of Reliability of Software at Stage of Acceptance Tests," PROGRAMMIROVANIYE, No 3, 1985, pp 77-81.

8. Karpovskiy, Ye. Ya. and Nemchinova, Ye. Ye. "Problem of Initial Data in Studies of Automated Control System Software," IZMERENIYE. KONTROL. AVTOMATIZATSIYA, Nos 1-2, 1980, pp 49-53.

9. Karpovskiy, Ye. Ya., Sagach, V. V. and Chernetskiy, A. A. "Nadezhnost algoritmov upravleniya" [Reliability of Control Algorithms], Kiev, Tekhnika, 1983, 112 pages.

10. Chizhov, S. A. "Designing Tests for Inventoried Software," USIM, No 6, 1984, pp 74-77.

11. Shneyderman, B. "Psikhologiya programmirovaniya. Chelovecheskiye faktory v vychislitelnykh i informatsionnykh sistemakh" [Programming psychology: Human Factors in Computer and Information Systems], Moscow, Radio i svyaz, 1984, 300 pages.

12. Klokov, Yu. K., Panushin, V. K. and Khamitov, R. R. "Methods of Improving Reliability of Software," ZARUBEZHNAYA RADIOELEKTRONIKA, No 6, 1984, pp 3-22.

13. Gorskiy, L. K., Karpovskiy, Ye. Ya. and Chizhov, S. A. "Evaluation of Control Tests for Inventoried Software," PROGRAMMIROVANIYE, No 6, 1985, pp 76-78.

14. Zhabeyev, V. P. "Simulator of Controlled Production System," PRIBORY I SISTEMY UPRAVLENIYA, No 10, 1982, pp 4-5.

15. Zhabeyev, V. P. and Filippov, V. Ye. "Generation of Test Sequences in Determination of Characteristics of Multichannel Measuring Information Systems" in "Sredstva nepreryvnogo kontrolya parametrov tekhnologicheskikh protsessov" [Facilities for Continuous Monitoring of Parameters of Production Processes], Kiev, UkrNPObumprom, 1983, pp 16-21.

16. Zhabeyev, V. P. and Nikolayev, V. P. "Automation of Processes of Estimation of Quality Indices of Real-Time System Software" in "Prikladnyye voprosy nadezhnosti programmnogo i tekhnicheskogo obespecheniya vychislitelnykh sistem" [Applied Questions Relating to Reliability of Computer System Software and Hardware], Kiev, KIIGA, 1985, pp 54-62.

17. Zhabeyev, V. P. and Minayev, Yu. N. "Use of Fuzzy Sets for Estimation of Reliability of Software" in "Nadezhnost i effektivnost ASUTP i ASUP: Tez. dokl. III Vsesoyuz. soveshchaniya" [Reliability and Efficiency of ASUTPs and Automated Enterprise Management Systems: Theses of Papers of Third All-Union Conference], Moscow, 1984, pp 49-50.

18. Minayev, Yu. N. "Prikladnyye metody otsenki kachestva funktsionirovaniya programmnogo obespecheniya vychislitelnykh sistem" [Applied Methods of Evaluating Quality of Functioning of Computer System Software], Kiev, Obshchesto "Znaniya" UkSSR, 1985, 21 pages.

19. "Systems Software. Results of Poll of Users," VYCHISLITELNAYA TEKHNIKA, No 34, 1985, pp 1-11.

UDC 681.3.06

# "Iset" Information Reporting System

18630234b Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan-Feb 88 (manuscript received 27 Mar 86, after correction 18 Dec 86) pp 107-109

[Article by V. G. Loginov, V. V. Pleshchev and V. V. Protasevich]

[Text] The Iset information reporting system was developed at the main collective-use computer center of the Sverdlovsk Oblast executive committee to meet the interactive information needs of users at various levels.

Level 1. These users are chairmen of executive committees and their deputies, chiefs of departments and administrations, and directors and their deputies. These people as a rule work with information that has already been organized into a report. The layout of these reports is quite varied, they contain integrated information concerning multiple plans, their retrieval time is minimal, and the interactions required to retrieve and review them must be as simple as possible. The reports for these users are prepared at the following levels.

Level 2. These users are functional specialists, not programmers, who have the background for generating reports. These users are capable of directly satisfying their own information needs and/or preparing information at the request of first-level users.

Level 3. These users are applications programmers who develop software for first- and second-level users. The Iset system offers third-level users support in the development and completion of tasks.

Level 4. These users are data processors who transfer the data to the Iset system for first- and second-level users.

Level 5. These users are the administrators of the Iset system. Administrators are responsible for the content of the database, its integrity, and reliability; and they grant other categories of users access to the database.

## Structure of Database

The database consists of documents for first-level users (completed reports), documents for second-level users (blank forms), documents for third-level users, and catalogs.

Completed reports are in the form of text or two-dimensional tables. Each report is characterized by the period of time over which it was formed and the date of storage. The date of storage is written in the text of the report. Completed reports are stored in data sets organized into a library or in sequential data sets with a record length of up to 128 bytes. If completed reports are stored in symbolic libraries, then each library contains documents of a single form and single kind of information. Each division of a library is made up of versions of a single document covering the various time period.

Blank forms are intended for the entry of information and/or for making the calculations necessary for the completion of the report.

Third-level user documents can be divisions of symbolic, object, or load libraries or data sets having any structure and organization.

A catalog is a description of a set based on some criterion. The database contains a user catalog, a section catalog, a topic catalog, a report catalog, and a time-period catalog.

### User Catalog

This is the highest-level catalog, and it contains the following information on each user: his code; the library name; section, topic, and report catalogs; the name of the volume in which the library is located; and, if necessary, the user's procedure for entering the system. Thus, user catalogs can be placed in multiple data sets, which prevents accidental damage to the catalogs and makes it possible to limit access to data.

### Section Catalog

This is the first catalog level in the Iset system that is relevant to the individual user. The organization of the subdivisions is indicated to assist the user in locating data. The sections are numbered sequentially, and information is given on the topics listed under each section.

### Topic Catalog

This is the second catalog level in the Iset system; it is subordinate to the section catalog. Topics are numbered sequentially in this catalog. If a topic is itself a section, then such is indicated in the topic catalog, and another topic catalog is listed under this topic. There are no limits to the branching of this hierarchy. The topics are the concrete manifestation of the organization of the division. In the specific case, a topic catalog is a catalog of scenarios or a previously-selected sequence of documents.

Report Catalog

This is the third catalog level that is relevant to users; it contains a listing of the specific reports available to the user on his chosen topic. Every topic that is not also a section or scenario contains a report catalog. The location of the report (volume, data set) and its formatting requirements (mode, width of margin, width of page, etc.) are all indicated in the report catalog.

Time-Period Catalog

If a report is stored in a library and it is indicated that it covers a specific time period, then the next catalog level is the catalog of time periods (or library divisions). This catalog is formed at the time of output of the report. The periods are numbered according to their dates, beginning with the earliest and ending with the latest. The divisions are numbered according to the order in which they appear in the library's index.

In order to ensure security and to limit access to the data in all catalogs stored in the database (section, topic, scenario and report catalogs), the codes of those users who have access to a specific section (topic or report) and for whom access is forbidden can be listed.

Structure of Software

The Iset system was developed to satisfy the needs of users at various levels for interactive access to a database, to ensure the integrity of the database over time, and to develop and manage the database. The Iset system includes four tasks for attaining these goals.

Iset-Input Task

This task addresses the goal of developing and managing a database. A catalog of users is put together and libraries with catalogs for each user are distributed (there can be a single library for all users or each user can have his own library). For third-level users working on the development and running of tasks it is desirable to assign each functional task its own library for catalogs.

Next, the range of questions about which the user will need information is formulated and a section catalog is constructed (e.g., industry section, territorial section). The range of topics on which information is to be offered is determined for each section (e.g., the topics under the industry section might be agriculture, urban transportation, construction, etc.). If a topic encompasses a wide range of questions, then it could be divided into subtopics (e.g., agriculture can be divided into animal husbandry, crop cultivation, etc.). If a topic is narrow, then it is divided into reports for which a report catalog is constructed (e.g., urban transportation can be divided into the following reports: entry of streetcars onto lines, entry of buses onto routes, etc.).

The next step is to determine the data sets in which the texts of reports will be stored. If the information is periodic in nature (e.g., daily, weekly, monthly, etc.), then a symbolic library with the library's divisions coded according to the date the document was created is preferable. A unique system for coding time periods was developed for this purpose. If the report stands alone and does not depend on the time of its creation, then it can be stored in a data set that is arranged sequentially. Documents for third-level users can be stored in data sets having any form of organization (sequentially indexed, direct, etc.). These sets and the method of processing them are indicated in the report catalog.

After the catalogs and data sets have been created, the method of loading each report is determined. The document can be created from a standardized report form, by means of the PRIMUS monitor [1], through batch processing and functional tasks, or from other reports through the Iset-Calculation facilities.

If a (symbolic) document is to be processed by the Iset-Archive task, it must have one of the three following control statements in its first line:

KHRANIT DO [STORE UNTIL] DD.MM.YY
KOPIROVAT S [COPY AFTER] DD.MM.YY or
UDALIT S [DELETE AFTER] DD.MM.YY

where DD, MM, YY [Day, Month, Year] is the date when the document must be either copied into the archive (in the first case with its deletion from the set, and in the second without its deletion) or deleted without copying (the third case). Catalogs and documents can be corrected either by means of the PRIMUS monitor or by means of the Iset-Interaction task.

Iset-Interaction Task

This is the principal task in the Iset system; it implements collective access to the database for all categories of users who have been permitted access to the Iset system (whose codes are in the catalog of users).

Entry into the task is attained by entering the "Iset" command on the screen of the PRIMUS monitor. If access to the Iset system is permitted, then the user conducts further work within the framework of the commands of this system.

A so-called initial scenario or sequence of commands affording immediate passage to the required level (e.g., the required catalog or document to be reviewed) can be established for each user in the Iset system.

All work in the Iset system is controlled through user commands. A listing of the section, topic, report, and time-period catalogs appears on the screen, from which the user can select the desired section, topic, report or time period by either its number or name. It is possible to obtain information about working in the Iset system at each working level. Having listed the sequence of numbers for the report, section, topic, and time period, the user

can proceed to either the review or correction level for the document, depending on which working mode was selected for the report from the report catalog. The operating mode can be changed at the time of output of the time-period catalog--for all periods at once or for a specific period. Of the available modes the following are of interest: document review, document correction (from spot corrections to complete rewriting), job execution (if the document is a batch-processing job written according to the rules of the Unified Series operating system), renaming of time periods (or divisions), and deletion and addition of new time periods (or divisions).

It is possible at any time to transfer to the PRIMUS monitor command mode (for third- and fourth-level users).

The interactive control commands are very simple and do not require special training. The following are the principal commands used at any stage of interaction.

N--Go to section catalog
A--Go to scenario catalog (only if a scenario catalog exists for the specific user)
T--Go to topic catalog
S--Go to time-period catalog
Ye--Return to preceding level
K--End session
?--Obtain instructions for working at the present stage of interaction.

Commands for going to catalogs only work going up the hierarchy, i.e., it is possible to go from the report catalog to the section catalog report using the N command; but in moving down the hierarchy it is necessary to indicate the number (or name) of the desired section, topic or report.

The scenario catalog contains the required sequence for accessing a document (or catalog). The required sequence is set into operation when the number of the scenario is indicated. Thus, the use of scenarios makes it possible to reduce to a minimum the sequence for finding a required (often used) document.

The Iset system provides the following functions for working at the document level:

The scanning of an entire document from beginning to end, page by page (by simply pressing the "Input" key).

Proceeding to a specified page (by entering the page number).

Going forward (back) a specified number of pages.

Finding a required page by context (the line on which the text to be checked lies can be specified in order to shorten the search time).

Correcting a page (by entering the I command).

Correcting a page in text blocks (the I'<tekst>' command). This command makes possible the standardized input of information in "windows," i.e., of preestablished divisions of a document for the entry of data. After one window is filled the cursor moves to another one, etc., until the entire screen is filled (all windows), or the contents of a window are not changed when the cursor arrives at it. Next, the data set is saved and "on-the-spot" updating continues.

Copying a page as the next page in the data set (the D command). The page can be duplicated as many times as is necessary, but it must be kept in mind that duplication is performed only within the framework of the data set (on-the-spot updating).

A single screen of readout is called a page. The number of lines on a page is indicated in the report catalog (no more than 22). In addition to the number of lines on a page, a document can have characteristics such as page headings (i.e., a group of lines that from the beginning of the document appear on every page), margin widths (if a document has a line length of more than 80 characters, then the right and left parts of the document can be viewed and data on the screen from the left half of the document can be saved), a position number for the second part of the document, etc.

In scanning it is possible to view a document for the previous date (the P-command), the next date (P+), the last date entered (P999), and the first (P1 or P0). If the document does not cover a time period, but is a division of a library, then the commands described above access the next, preceding and other divisions according to their position in the library index.

Iset system administrators are afforded the capability of scanning the database by indicating the code of a specified user in the interaction process. They can thus keep track of the database's integrity and monitor the security system. A system administrator can also promptly intervene in the actions of other users, scan a specified user's screen buffer, and remove his Iset-Interaction task if necessary.

The gathering of statistics on a user's work at a terminal is provided for in the Iset-Conversation task. Each command by a user is recorded in a specially organized data set for gathering statistics which can be printed by the system administrator as necessary. The keeping of statistics makes it possible to keep track of users' work and of access to the database. It is also possible to write scenarios for specific user groups on the basis of statistical data.

Iset-Archive Task

This task is used for keeping an archive of "obsolete" documents; and it performs the tasks of loading obsolete documents (with expired storage dates) into the archive (superfluous gaps are deleted in the process), dumping documents from the archive on request, and forming an archive inventory list.

The database's archives are managed user by user, which makes it possible to distribute the various user-level documents in various archives. Third-level users can also manage archives by themselves, while management of the archives for first- and second-level users can be entrusted to the Iset system administrators.

For programmer users the archive system makes it possible to keep old versions of their programs, data, etc. They can thus return at any time to an earlier operating version of a program.

Iset-Calculation Task

This task is designed for automating the performance of calculations on documents stored in the database (blank forms and completed reports).

Blanks are prepared by the OFORT system [2]. A form consists of various groups of lines--some intended for input data and others for output. It is input by batch processing or interaction with the computer. Windows, i.e., places for the input of variable data, are included in these groups of lines. The windows are identified by labels that uniquely indicate their address. All calculations are indicated on the form at the label level. The following operations are permissible: the calculation of percentages and other arithmetic operations, the transfer of numeric and symbolic data, totaling, and decoding.

Variable data can be entered in the windows from the Iset-Interaction task by means of the correction-by-context command or by filling in the form using functional tasks.

A single form can be used for many different documents that have the same format, i.e., for which the location of the windows and the calculations are identical.

A calculation command is required for each form. This command is a procedure describing in fairly simple language the sequence of operations for completing the form, i.e., what documents serve as input for calculations, to which groups of lines they apply, and under what name and location the output information is to be placed. If it is necessary to file the output report according to the Roman or Russian alphabet, then the sorting procedure is also indicated in the calculation command.

Functional Characteristics of the Iset System

Let us present some data on the Iset-Interaction task. Main memory capacity required is 8K + 6K x N (where N is the number of users working simultaneously). Reaction time is 0.5 to 1 s when going from level to level. If a report search scenario is initiated, then the reaction time is 2 to 3 s (for a YeS7927 local display station and YeS1033 computer; the reaction time is two to three times longer for a remote display station).

## BIBLIOGRAPHY

1. Vasilkov, V. V., Yelomanova, V. V. and Ivanov, V. N. "Organization and Development Prospects for an Interactive System" in "Proyektirovaniye sistem kollektivnogo polzovaniya vuza" [Designing VUZ Collective-Use Systems], Moscow, Atomizdat, 1980, pp 15-31.

2. Pleshchev, V. V. and Borovskikh, N. N. "Tabulation Designer," USIM, No 1, 1983, pp 75-76.

UDC 338:681.3

# Software Tools for the Generation of Reports Using the Language DLR and the PALMA DBMS

18630235a Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan-Feb 1988 (manuscript received 11 Mar 86; after revision 18 Dec 86) pp 65-68

[Article by V. G. Frolov]

[Abstract] This article examines methods and software for the generation of reports using the high-level language DLR provided with PALMA, the first Soviet relational data base system. The major software facility provided is the RELPRINT program, which accesses a PALMA data base, performs page and line formatting of selected data and prints out a report based on parameters contained in a library of parameters written in the language DLA. These report parameters can be calculated manually or automatically generated by the NEATAB3 program. The RELPRINT program consists of the following modules: PROC RELPRINT (prints the report), QUERYGET (queries the data base), HEADRDR (places headers on the page), PARMGET (reads report parameters), PLMEXEC (reads one line), ANALYZE (analyzes the line), a line printing routine and PRINT-PAGE (prints the last page of the report). The individual modules of the RELPRINT program are briefly described. Due to its modular structure, the RELPRINT program can serve as a basis for the generation of a wide variety of reports in the PALMA DBMS. References 8: Russian.

UDC 621.382.82.001

ARIS Circuit Modeling System

18630235b Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan-Feb 1988 (manuscript received 2 Jun 86; after revision 4 Nov 86) pp 94-96

[Article by B. V. Batalov, S. G. Rusakov, V. P. Batagin, M. M. Zharov, A. A. Lyalinskiy, and S. L. Ulyanov]

[Abstract] The ARIS (acronym for automated design of integrated circuits) system models the electrical characteristics of LSI circuits based on bipolar and metal-dielectric-semiconductor devices. Various versions of ARIS run on the YeS and SM computers, the Elektronika-82 superminicomputer and the DVK-2M microcomputer. All versions share a common structural, linguistic, algorithmic and mathematical basis. This article discusses the capabilities of the software, including expansion of the system with additional eleemtn models and types of analysis, hierarchical description of circuits provided by a fragmentation apparatus with no limits as to depth of embedding and number of fragments at each level, and generation of a metal-dielectric-semiconductor transistor list in the form of sublists of topologic and technological information. The structure and operating algorithms of the system and the functional capabilities of the various versions are described. The YeS computer version allows for a maximum of 500 components, 250 nodes, 100 model parameter lists and 250 fragments. The memory requirement is 310 Kbytes. References 12: 8 Russian, 4 Western.

UDC 681.3.06:621.3.049.774.3'14.001.2

Data-Processing Software for Manufacture of VLSI Photographic Templates

18630241 Moscow PRIBORY I SISTEMY UPRAVLENIYA in Russian No 2, Feb 88, pp 14-15

[Article by V. A. Lementuyev and V. Z. Popov, engineers]

[Abstract] The machine description of VLSI topology produced by a CAD system is in the form of a set containing the geometric fields of each layer of the circuit. This article describes the TEMOS software package, which converts this machine-description of VLSI topology to a description of the photographic templates required to manufacture the physical circuit. The components of the TEMOS package and their basic functions are: the PUROV program makes the transition from contour definition of topology to an ordered vector-level representation; the PUNION program implements the set-theory operations required to generate the individual layers; the PREOBR program makes the transition from vector format to the SOURCE or PPV format; the PSORT program performs global sorting of the topology file for a layer or the template description file; the PCOPY program performs

file transfer between the KULON-1 and RAFOS operating systems. The total length of the package is over 13,000 FORTRAN commands. Time required for performance of the individual operations in the TEMOS package and the length of the source and output files are presented for four topology fragments of varying complexity, indicating that the program package achieves a reduction in file length by approximately a factor of two. Figures 1, tables 1, references 2: Russian.

UDC 658.512.011.56:622

## Mining Technology a CAD System for Open-Pit Mines

18630236a Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan-Feb 88 (manuscript received 3 Feb 87) pp 79-82

[Article by V. A. Kovalenko and E. B. Sakimbayeva]

[Abstract] The traditional method of manually calculating the parameters for blasting operations at open-pit mines has a number of shortcomings: it is extremely labor intensive, it requires the calculation of correctional tables for the parameters of the explosive charges, and it inevitably involves errors in determining the geometry of the site and the coordinates of the bore holes. A significant improvement in the quality of blasting operations can be achieved by the use of modern planning systems and computers. A computer-aided design (CAD) system is suggested for automatic planning of drilling and blasting operations. The system consists of three major subsystems--mine surveying, blast planning and graphic output. The system operates with a data base including data from survey operations and geological studies at the mine. A flow chart and diagrams illustrate application of the system to planning of a typical open-pit mine blast. Figures 4, references 3: 2 Russian, 1 Western.

UDC 681.3.513

## Determination of Contours in a Graphic Data File

18630236b Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan-Feb 88 (manuscript received 8 Jan 85; after revision 23 Jul 87) pp 88-89

[Article by A. K. Gruzdev]

[Abstract] For the purposes of this article, a graphic data file is assumed to consist of straight line segments, circles, and circular arcs. The article focuses on the task of determining the contours of all the fields into which a plane is divided. In the first stage, the file is divided up into elements

whose end points are the intersection points of the graph. In the second stage, any superfluous coinciding elements are eliminated. In the final stage, the actual contours are found by first replacing each element of the graph with two opposing directed line segments or arcs. A contour corresponds to one cycle in the oriented graph that results from this step. This method of determinig contours places no restrictions on the order of the elements within the graphic data file or their coordinates on the drawing. The method has been implemented in FORTRAN for the SM-4 computer as a part of a CAD system. Figures 2, references 3: Russian.

UDC 681.324

Comparative Assessment of Multifunctional and Special-Purpose Network Information Systems

18630222a Riga AVTOMATIKA I VYCHISLITELNAYA TEKHNIKA in Russian No 1, Jan-Feb 88 (manuscript received 18 Feb 87) pp 43-48

[Article by E. V. Zinovyev]

[Abstract] Present-day information systems can be classified as either multifunctional (general-purpose) or special-purpose, depending on their organization and the problems solved. General- and special-purpose features can now be combined in complex computer systems all functional levels, including operating systems, programming languages, application programs and hardware. The development of network information systems involves selecting the structure and make-up of software and hardware. It thus becomes necessary to assess the efficiency of both multifunctional and special-purpose systems.

An analysis is presented of the two kinds of organization of network information systems--special-purpose (computer system No 1) and multifunctional (system No 2). Queuing theory and reliability theory are empolyed to determine the maximum throughput and reliability of the two systems. The areas of application in which multifunctional systems are more efficient than special-purpose ones are determined. It is assumed that systems 1 and 2 are equivalent with respect to hardware configuration and user requests and that they are comprised of an equal number of computers of the same type. But the internal organization of the two systems differs. Both systems offer the user access to the set R of various processes and resources. System 1 consists of specialized single-function computers, each of which supports only one process or resource from st R. System 2, however, contains n multifunctional computers, each of which can support all processes and resources of set R.

It is assumed that a multidimensional nonstationary Poisson input of user requests enters both systems and the rate of input of a given type of request is a known function of time. Maximum throughput serves as the efficiency criterion. Each computer is regarded as a service machine and the length of the request queue is unlimited. Both systems contain a control element

(dispatcher). It is demonstrated that system 2 is capable of more fully utilizing the throughput of its computers when request inputs are non-stationary. It is also shown that the efficiency of a computer system as measured by throughput and reliability, tends to increase as it passes from specialized to multifunctional organization. This gain is at the expense of further complication of the system's internal structure and process control algorithms. Generally, multifunctional systems provide a higher level of efficiency for computer systems. There are cases, however, in which special-purpose systems can be favored in view of the relative simplicity of their organization. Such a case occurs when the throughputs of systems 1 and 2 are equal--when inputs of various types are balanced, i.e., the rate of input is at a maximum for system 1 and is completely consistent with the computer's throughput capabilities. Figures 4; references 6: Russian.

UDC 681.323

Adaptive Memory Management

18630222b Riga AVTOMATIKA I VYCHISLITELNAYA TEKHNIKA in Russian No 1, Jan-Feb 88 (manuscript received 13 Jul 87 (17 Feb 87)) pp 49-56

[Article by A. A. Prikhozhiy]

[Abstract] The problem of minimizing the exchange of information between various levels of memory in virtual page-oriented memory systems is discussed. Response time is shortened and the computer system's throughput is increased when information exchange is reduced. The pages used around a certain time, t, comprise a "local set," which varies over time. Memory management would be best if the pages from the local set that will be referenced in the future were in main memory. Since it is not possible to know the local set precisely, an estimate of it, Z(t), called the working set, is used in practice.

The basic task of memory management is to estimate the working set in the process of running a program. Two types of memory management strategies are used to complete this task. The first assumes that the power of the working set varies over time and the other assumes that this power is constant. The most well known strategy, which is of the first type, is Denning's strategy. A number of modifications of Denning's strategy are also used. The most well known strategy of the second type is LRU. Denning's LRU, FIFO and MIN strategies are all summarized. It is demonstrated that Denning's strategy and other familiar strategies are particular cases of the adaptive strategies that are presented here. These adaptive strategies can be adjusted for specific programs and are capable of procedural changes in the middle of a run. The theory of adaptive memory management is presented and the Ap and $A_m$ strategies are summarized. The $A_p$ strategy is oriented toward programs with variable power of the local set; it minimizes the amount of main memory occupied and keeps constant the probability of

referral to external memory by varying when necessary the number of pages in main memory. The $A_m$ strategy is oriented toward programs with constant power of the local set; it places a fixed number of pages in main memory. The $A_m$ strategy is compared with the LRU, FIFO and MIN strategies.

The results are given of a computing experiment that simulates the various strategies. The strategies were evaluated on the basis of the frequency of page exchange with external memory. The $A_m$ strategy made possible an exchange frequency 144 percent lower than that of the MIN strategy, 23 percent lower than that of the LRU strategy and 20 percent lower than that of the FIFO strategy. The $A_p$ strategy worked on the whole better than the $A_m$ strategy and made possible an exchange frequency 9 percent lower than that of Denning's strategy. Adaptive strategies are not markedly worse than the familiar strategies in terms of complexity of implementation and speed and are therefore promising for practical utilization. Figures 2; references 11: 8 Russian, 3 Western.

UDC 681.324

Information-Computer Networks Based on Radio Communications

18630232 Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan-Feb 88 (manuscript received 27 Apr 87) pp 37-43

[Article by S. G. Bunin and A. M. Luchuk]

[Abstract] Computer networks utilizing radio transmission rathern than cables have many advantages including great economy, high speed and ease of network development and modernization. Equations are presented for computation of the number of channels required and mean load on each frequency in a packet-switching computer network. The experience of the ALOHA network of the University of Hawaii is cited to demonstrate the advantages of a common relay over a central-computer to facilitate data interchange among numerous, sometimes mobile, users. The interaction of cells in a cellular radio network is discussed. Interaction of cells through sluices and geosynchronous satellites according to ISO protocols is noted as a possibility. Computer networks utilizing radio channels as their transmission medium have a number of unique properties that make them irreplaceable in networks with large numbers of often mobile users where costs must be controlled. Even local networks and computer systems can utilize radio channels and digital radio signals for high-speed data exchange, thus making possible the creation of computing environments with a flexible architecture that is capable of changing during the performance of a task. Figures 9, references 12: 11 Russian, 1 Western.

- END -

T0

22161
55
NTIS
ATTN: PROCESS 103
5285 PORT ROYAL RD
SPRINGFIELD, VA 22161



Foreign Broadcast Information Service (FBIS) and Joint Publications Research Service (JPRS) publications contain political, economic, military, and sociological news, commentary, and other information, as well as scientific and technical data and reports. All information has been obtained from foreign radio and television broadcasts, news agency transmissions, newspapers, books, and periodicals. Items generally are processed from the first or best available source; it should not be inferred that they have been disseminated only in the medium, in the language, or to the area indicated. Items from foreign language sources are translated; those from English-language sources are transcribed, with personal and place names rendered in accordance with FBIS transliteration style.

Headlines, editorial reports, and material enclosed in brackets [ ] are supplied by FBIS/JPRS. Processing indicators such as [Text] or [Excerpts] in the first line of each item indicate how the information was processed from the original. Unfamiliar names rendered phonetically are enclosed in parentheses. Words or names preceded by a question mark and enclosed in parentheses were not clear from the original source but have been supplied as appropriate to the context. Other unattributed parenthetical notes within the body of an item originate with the source. Times within items are as given by the source. Passages in boldface or italics are as published.

## SUBSCRIPTION/PROCUREMENT INFORMATION

The FBIS DAILY REPORT contains current news and information and is published Monday through Friday in eight volumes: China, East Europe, Soviet Union, East Asia, Near East & South Asia, Sub-Saharan Africa, Latin America, and West Europe. Supplements to the DAILY REPORTs may also be available periodically and will be distributed to regular DAILY REPORT subscribers. JPRS publications, which include approximately 50 regional, worldwide, and topical reports, generally contain less time-sensitive information and are published periodically.

Current DAILY REPORTs and JPRS publications are listed in *Government Reports Announcements* issued semimonthly by the National Technical Information Service (NTIS), 5285 Port Royal Road, Springfield, Virginia 22161 and the *Monthly Catalog of U.S. Government Publications* issued by the Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.

The public may subscribe to either hardcover or microfiche versions of the DAILY REPORTs and JPRS publications through NTIS at the above address or by calling (703) 487-4630. Subscription rates will be provided by NTIS upon request. Subscriptions are available outside the United States from NTIS or appointed foreign dealers. New subscribers should expect a 30-day delay in receipt of the first issue.

U.S. Government offices may obtain subscriptions to the DAILY REPORTs or JPRS publications (hardcover or microfiche) at no charge through their sponsoring organizations. For additional information or assistance, call FBIS, (202) 338-6735,or write to P.O. Box 2604, Washington, D.C. 20013. Department of Defense consumers are required to submit requests through appropriate command validation channels to DIA, RTS-2C, Washington, D.C. 20301. (Telephone: (202) 373-3771, Autovon: 243-3771.)

Back issues or single copies of the DAILY REPORTs and JPRS publications are not available. Both the DAILY REPORTs and the JPRS publications are on file for public reference at the Library of Congress and at many Federal Depository Libraries. Reference copies may also be seen at many public and university libraries throughout the United States.